International Journal of Mechanical and Production Engineering Research and Development (IJMPERD)
ISSN (P): 2249-6890; ISSN (E): 2249-8001
Vol. 7, Issue 3, Jun 2017, 173-194

MULTI-OBJECTIVE FOR A PARTIAL FLEXIBLE OPEN SHOP SCHEDULING PROBLEM USING A PRIORITY-BASED EVOLUTIONARY ALGORITHM

N. JANANEESWARI[1], S. JAYAKUMAR[2] & M. NAGAMANI[3]

[1]Research Scholar of Mathematics, Aringar Anna government Arts College, Cheyyar, Tiruvannamalai, India

[2]Assistant Professor & Head, Department of Mathematics, Aringar Anna Government Arts college, Cheyyar, Tiruvannamalai, India

[3]Associate Professor of Mathematics, Global Institute of Engineering & Technology, Vellore, India

ABSTRACT

In this study, an evolutionary algorithm (EA) with priority-based representation has been applied to get a partial flexible, open shop scheduling problem (PFOSP) which happens to be one of the hardest combinatorial and operations research problems. The priority for each operation is represented by a genetic on a best chromosome by a constructive algorithm performed for the decoding method on all active schedules, which will constitute for a subset of partial feasible schedules including the optimal solution. To obtain improved solutions, iterative local search algorithm (ILSA) is applied to the best chromosome to obtain at the end of each reproduction process. The most widely used PFOSP data sets are generated in the literature which are used for benchmarking and evaluating the performance of the proposed EA methodology. The computational results show that the proposed EA performed at the same level or at a better level with respect to the minimization of make span for some data sets when compared with the results based from the literature.

KEYWORDS: Evolutionary Algorithm (EA), Open Shop Scheduling, Minimize Make Span, Priority-Based, Local Search & Non Dominated Pareto Optimal

Received: Mar 20, 2017; **Accepted:** Apr 17, 2017; **Published:** May 25, 2017; **Paper Id.:** IJMPERDJUN201720

Original Article

INTRODUCTION

Scheduling theory is a major significant research area in Operations Research. It has been the subject of research which ranges from simple dispatching rules to the sophisticated learning algorithms [39]. Open shop scheduling problem (OSSP) is the most studied topic in sequencing and scheduling theory which is known to be much difficult to solve at. Although the name "open shop scheduling" seems that it is an industrial problem, the structure of OSSP is much more relevant to various applications, such as management, computing, and also in public services. The detailed review of OSSP can be had from the references such as Jain and Meeran [39] and Lee and Pinedo [46].

In OSSP, there are n jobs planned and scheduled on m machines. Each job will comprise of consecutive operations, which are precedence relations to be filled. Each operation is processed by a particular machine at a particular time.

The partial flexible open shop scheduling problem (PFOSP) is a complex problem and can be generally

compared to classical OSSP. Unlike the OSSP, each operation can be preceded by any one of the machines from a given machine set. The objective of PFOSP is to minimize for both the assignment and a corresponding scheduling which will minimize the make span time. The rest study on FOSP was performed in 1990 by Brucker and Schlie [12] who developed a polynomial time algorithm for FOSP with minimum two jobs. The approximation algorithms have been developed in FOSP literature to get the best results for large size instances [65]. Furthermore, progressive hybrid methodologies have also been developed to study the performance of existing techniques.

In FOSP, an NP-hard problem is much stronger than OSP even though they have much more similarities [31] which in challenge to obtain good results for FOSP in a reasonable run time. Evolutionary algorithms (EA), is a well known meta-heuristic algorithms, which have received remarkable interest in addressing the FOSP. Oliveira et al. [62] used random key representation schema for OSP to represent the priorities in the Giffler and Thompson Algorithm (GTA), is also the priori which is a constructive algorithm developed by Giffler and Thompson in 1960 for OSPs [32]. By this method, the search area containing of only active schedules which are represented by the generated best chromosome. The chromosome structure is used to` see the effect of the representation on FOSP. A constructive algorithm based on GTA, which can produce all active schedules for FOSP, is used for decoding. According to the representation used by Oliveira et al. [62], a solution can be represented by multiple chromosomes. In this study, permutation encoding is used to reduce the number of alternative chromosomes representing the same solution. In order to improve the diversity and quality of the chromosomes, problem based operators are used for FOSP.

The organization of this paper is as follows. In the next section, a brief presentation of FOSP is given. A literature review on various algorithms developed for FOSP is presented in the proposed priority-based EA. Computational results are presented for well-known instances from the literature. The last section has been reserved for the conclusion, discussions and suggestions for further studies.

The partial flexible open shop scheduling problem: The FOSP consists of n jobs J_1, J_2,,,,,, J_n and m machines M_1, M_2,,,,, M_m including different tools. Each job J_i contains n_i operations, O_{i1}, O_{i2},,,,,,, O_{ini}. There are sets of M_{ij} for M_1,,,,, M_m for each operations of O_{ij}(i = 1,2,,,, n; j = 1,2,,,,, n_i) which has to be assigned on a machine in the set for being processed without preemptive. The p_{ijk} is the processing time of operation O_{ij} on the machine k. Moreover, on a maximum one operation can be processed on a machine at the same time and the specified job cannot be processed on more than one machine at the same time [12].

The two jobs and three machines, Operation O_{11} can be processed on M_1 or M_2, O_{12} and O_{21} can be processed on M_2 or M_3 and O_{22} must be processed on M_3. An assignment is feasible, if (O_{ij}) M_{ij} for all i = 1,2,,,,, n and j = 1,2,,,,,, n_i. A schedule is the demand on a performance measure for the assignment. A schedule is feasible only if it is feasible for a OSP based on an assignment [12]. When M_{ij} = 1 for all O_{ij}, the problem becomes a OSP. Because of the existent alternative machines to process one operation, the FOSP is more complicated than the OSP [11]. Only the sequence of the operations is decided in the OSP. The assignment of operations to the machines is mixed. On the other hand, both the assignments and the order of operations on each machines are decided in the FOSP [66]. Other performance measures typically used in this literature study are minimum time, total workload, critical machine workload, total tardiness and mean tardiness. In the three objective notation schemes [34], the problem taken for our study is denoted by FOSP C_{max} where FO denotes FOSP and C_{max} represents make span.

LITERATURE REVIEW

The studies developing approximation algorithms for FOSP are investigated and widely considered in our case study. The optimization methods used in FOSP literature are the parameters used in this study. The studies on single and multi-objective FOSPs and the methodologies utilized to solve the corresponding problems are also considered. EA is one of the well known population based algorithms [8] which is also most widely used approximation method in FOSP literature. The major part of these studies combines EA with a local search algorithm to utilize both the global search and the local search abilities. Gao et al. [27] hybridized the EA and bottleneck shifting procedure for multi objective FOSP. Zribi et al. [88] solved the FOSP by considering assignment and sequencing problems hierarchically. Two methods, based on local search and the branch and bound algorithm, were developed for the assignment problem while a hybridized EA was proposed for the sequencing. Gao et al. [27] combined EA with two neighborhood and friendly structures to develop in a local search to solve the FOSP with non-mixed availability constrains. Gao et al. [29] also combined EA with variable neighborhood search to increase in their search ability.

Table 1: Algorithms Used in the FOSP Literature

S. No	Short Form	Abbreviation Algorithm
1.	ACO	Ant Colony Optimization
2.	AIS	Artificial Immune System
3.	AL	Approach by Localization
4.	BBA	Branch and Bound Algorithm
5.	BSP	Bottleneck Shifting Procedure
6.	CPM	Critical Path Method
7.	DEA	Differential Evolution Algorithm
8.	DFA	Discrete re y algorithm
9.	FBS	Filtered beam search
10.	EA	Evolutionary Algorithm
11.	GRASP	Greedy Randomized Adaptive Search Procedure
12.	HAS	Harmony Search Algorithm
13.	IAM	Insertion Algorithm method
14.	ILS	Iterated Local Search
15.	LNS	Large Neighborhood Search
16.	LSA	Local Search Algorithm
17.	PDR	Priority Dispatching Rules
18.	PSO	Particle Swarm Optimization
19.	SA	Simulated Annealing
20.	TS	Tabu Searh
21.	VNS	Variable Neighborhood Search

Kacem et al. [42] developed an assignment and scheduling the procedure, known as approach by localization. In order to improvise neighborhood results to overcome real life problems, they hybridize the EA with the approach on localization to combine the gain of the methods. Pezzella et al. [66] integrated different strategies to improve the performance of EAs of the FOSP. They used the approach by localization to generate the initial solution and dispatching the rules to get the sequencing of the initial assignments. Li et al. [**50**] developed an hybrid method for combining the variable neighborhood search and the EAs to solve the multi-objective FOSP. Frutos et al. [25] used the EA with simulated annealing to join local and global search for solving multi-objective FOSP. Wang et al. [77] and Gao et al. [30] used a EA which is related on immune and entropy principles to solve the multi-objective FOSP.

Al-Hinai and ElMekkawy [1] combined an heuristic used for their initial population generation, a local search

method, and a EA for FOSP. Moradi et al. [58] developed and compared the four multi-objective optimization methods, which will combine the EA and priority for dispatching rules to reduce the make span and system unavailability, for the integrated FOSP with non-mixed or combined preventive maintenance activities. Xing et al. [80] proposed a multi-population interactive co-genetic algorithm to obtain and increase the performance of evolutionary algorithms of the FOSP. Both the artificial ant colonies and EA with different conjurations were applied to establish each of the population independently and is the interaction; competition and mechanism of sharing among populations were satisfied. Ho et al. [37] proposed a new coding technique for a learnable EA which promotes an effective integration among evolution and learning.

Table 2: Single Objective FOSP Studies

Study	Year	Problem	Methodology
Baykasoglu [6]	2002	FOSP	SA + PDR
Schrich et al. [71]	2004	FOSP	TS + PDR
Ho et al. [37]	2007	FOSP	EA + PDR
Liouane et al. [52]	2007	FOSP	TS + AC
Fattahi et al. [24]	2007	FOSP	TS + SA
Pezzella et al. [66]	2008	FOSP	EA + AL + PDR
Moradi et al. [57]	2010	FOSP	EA + PDR
Bozejko et al. [10]	2010	FOSP	TS + IA
Xing et al. [80]	2011	FOSP	EA + AC
Al-Hinai and ElMekkawy [2]	2011	FOSP	EA + LS
Karimi et al. [44]	2012	FOSP	VNS
Roshanaei et al. [70]	2013	FOSP	AIS + SA
Yuan and Xu [82]	2013	FOSP	DEA + LS
Wang et al. [78]	2013	FOSP	EA
Farughi et al. [23]	2013	FOSP	EA + CPM
Yuan and Xu [83]	2013	FOSP	HSA + LNS
Yuan et al. [85]	2013	FOSP	HSA + LS
Mousakhani [60]	2013	FOSP	ILS
Ziaee [88]	2014	FOSP	Meta-Heuristic
Driss et al. [20]	2015	FOSP	EA

Within a searching process in random, Moradi et al. [57] used a learnable EA to prove FOSP with preventive maintenance activities and the composite dispatching rule to produce population. Tay and Ho [75] used a composite dispatching rule generated for Evolution programming. They recognized that no rule has remarkable through put on all criteria and combinations for the rules to enhance the efficiency of the rules and quality of the results. Farughi et al. [23] studied for hybridized EA and a meta- heuristic related on critical path method for PFOSP with their overlapping operations.

The meta-heuristic is used to improve the results of the proposed methodology. Chiang and Lin [17] proposed a genetic algorithm for a multi-objective PFOSP. The distinctive characteristics of the proposed algorithm for its simplicity are based on two parameters, effective evolutionary operators and population diversity. Yuan and Xu [83] also proposed new memetic algorithms for hybridizing EA and local search for the multi-objective PFOSP. Driss et al. [20] developed a EA with novel representation for novel and operations research for FOSP reducing the make span. The proposed algorithm is to validate on the series of benchmark data sets and tested on the data received from a drug manufacturing company. Wang et al. [78] proposed a novel chromosome encoding for the PFOSP and which uses the time machine assignment and

adapting to scheduling rules to achieve the robustness and eligibility. Rahmati et al. [67] developed non-dominated sorting of EA and non dominated ranking EA for multi-objective PFOSP and he proposed new multi-objective Pareto-based modules and a new measure for the multi-objective evaluation.

Table 3: Multi-Objective FOSP Studies

Study	Year	Problem	Methodology
Kacem et al. [42]	2002	FOSP	EA + AL
Baykasoglu et al. [7]	2004	FOSP	TS + PDR
Xia and Wu [79]	2005	FOSP	PSO + SA
Gao et al. [26]	2006	FOSP	EA
Gao et al. [27]	2007	FOSP	EA + BSP
Zribi et al. [89]	2007	FOSP	EA + BBA + LS
Gao et al. [28]	2008	FOSP	EA + VNS
Tay and Ho [75]	2008	FOSP	EA + PDR
Wang et al. [76]	2008	FOSP	FBS + PDR
Zhang et al. [87]	2009	FOSP	PSO + TS
Li et al. [50]	2010	FOSP	EA + VNS
Frutos et al. [25]	2010	FOSP	EA + SA
Wang et al. [77]	2010	FOSP	EA + AIS
Gao et al. [30]	2010	FOSP	EA + AIS
Grobler et al. [35]	2010	FOSP	PSO + PDR
Li et al. [48]	2010	FOSP	TS + VNS
Moradi et al. [58]	2011	FOSP	EA + PDR
Moslehi and Mahnam [59]	2011	FOSP	PSO + LS
Li et al. [49]	2011	FOSP	PSO
Li et al. [47]	2011	FOSP	PSO
Rajkumar et al. [68]	2011	FOSP	GRASP
Chiang and Lin [17]	2013	FOSP	EA
Rahmati et al. [67]	2013	FOSP	Gas
Shao et al. [72]	2013	FOSP	PSO + SA
Gao et al. [29]	2014	FOSP	HSA + LS
Jia and Hu [41]	2014	FOSP	TS
Karthikeyan et al. [45]	2014	FOSP	DFA + LS
Li et al. [51]	2014	FOSP	PSO + TS
Rohaninejad et al. [69]	2015	FOSP	EA
Yuan and Xu [84]	2015	FOSP	EA + LS

Rohaninejad et al. [69] proposed a nonlinear IP model and also the hybridized EA with meta-heuristic, which is a multi-attribute decision making method, for multi-objective PFOSP with machines capacity constraints. The computational results are obtained by well-known multi objective algorithms from the literature showed that the proposed algorithm to obtain throughout better performance, especially in the closeness of the solutions result to the Pareto optimal front. The particle swarm optimization algorithm is another alternate method which was widely used for approximation method in PFOSP literature. Xia and Wu [79] combined simulated annealing with particle swarm optimization to omit, being trapped in a local optimum for multi-objective PFOSP. Grobler et al. [35] developed four particle swarm optimization similar to meta-heuristic approach for multi-objective PFOSP with sequence-dependent setup times. The priority-based particle swarm optimization algorithm was executed for the best performance and respect to the quality of a solution and their computational complexity. Zhang et al. [86] hybridized the particle swarm optimization algorithm with tabu search algorithm, which was used as the local search strategy for each and every particle for the multi-objective PFOSP. Moslehi and Mahnam [59] combined and joined the particle swarm algorithm and a local search algorithm for multi-objective

PFOSP with various release times. Shao et al. [72] developed a hybrid methodology, which uses the discrete particle swarm optimization for their global search and simulated annealing for local search and for multi-objective PFOSP. Li et al. [47] and Li et al. [49] proposed a hybridized artificial bee colony algorithm, which is a novel particle swarm methodology, for solving the multi-objective PFOSP. Li et al. [51] combined artificial bee colony algorithm with tabu search for the multi-objective PFOSP with maintenance activities. They used a self-adaptive strategy to generate neighboring solutions in different promising regions and also designed an external Pareto archive set to record the obtained non-dominated solutions.

Many studies developed methodologies was depended on the tabu search algorithm because the tabu search algorithm exhibits an efficient local search. Schrich et al. [71] performed two methodologies, a hierarchical and multi-start tabu search, using the priority dispatching rules to obtain the initial solution. Fattahi et al. [24] compared integrated and hierarchical solution approaches for PFOSP. The computational results show that the hierarchical algorithms have better performance and output than that of the integrated approach. Among six variant hybrid searching methodologies, including the searching approach and heuristics, the algorithm combining the tabu search and simulated annealing algorithms for the assignment and sequencing sub problems consecutively performs better than the other algorithms. Bozejko et al. [10] developed a two module algorithm which contains of a machine selection and the operation scheduling for solving PFOSP. The insertion algorithm and the tabu search algorithm with backtracking, which was taken from the literature, where it can be applied for the operation scheduling module. It is inferred that the exact algorithms can be used on both of the modules to obtain an optimal solution. Li et al. [48] was a hybrid algorithm with the two modules to handle the multi-objective PFOSP. In the machine assignment module, a tabu search algorithm was used to generate neighboring solutions while a variable neighborhood search algorithm is applied to the local search in the operation scheduling module. Li-ouane et al. [52] combined ant colony optimization meta-heuristics with local search methods, which consists of tabu search, along with the efficiency of the local search methods with an ant system approach. Baykasoglu [6] and Baykasoglu et al. [7] used grammar-form linguistics to represent the data and the dispatching rules to arrange and manage for the operations. Baykasoglu [6] used a meta-heuristic method related to simulated annealing for optimization though Baykasoglu et al. [6] used the tabu search algorithm for multi-objective optimization. Jia and Hu [41] proposed a path re- linking algorithm related to the tabu search algorithm with the back jump tracking to prove the multi-objective PFOSP. Wang et al. [75] developed altered beam search based heuristic algorithm to solve multiple-objective PFOSP. They use the dispatching rules based heuristics as local and global evaluation functions to omit useless paths and reduce the computational time. Zee [88] developed a heuristic algorithm for PFOSP minimizing the make span which can obtain high quality solutions in very short time duration. Rajkumar et al. [68] proposed a GRASP algorithm to prove the multi objectives of FOSP to be solved with limited resource constraints. Karimi et al. [44] proposed a knowledge-based variable neighborhood search algorithm is to search the solution space for neighborhood solutions, extracts the knowledge of good solution and feed it back to the algorithm. Mousakhani [60] proposed a MIP model and a meta-heuristic algorithm based on iterated local search for the PFOSP with the sequence dependent setup times to reduce the total tardiness. The proposed model already has better results in terms of both size and computational complexities than to the model from the literature. Roshanaei et al. [70] developed the two MIP models for PFOSP to reduce the total make span. To solve the large size instances, meta-heuristic hybridizing artificial immune and simulated annealing algorithms are developed. The proposed hybrid algorithm is compared with the others from the literature using benchmark instances and an industrial problem of a mould and dies manufacturing company. Yuan et al. [85] proposed a hybrid harmony search algorithm integrated with the local search

procedure for solving the PFOSP to minimize the total make span. Furthermore, according to Yuan and Xu [83], combination of hybrid harmony search with large neighborhood search and Yuan and Xu [82] also proposed hybrid differential evolution algorithms for the similar problem. Gao et al. [29] used a discrete harmony search algorithm to prove the multi-objective PFOSP. They used the local search to initiate the harmony memory and enhance the exploitation capability. Karthikeyan et al. [45] combined hybridized discrete delay algorithm with local search to enhance the exploitation capability.

In the inspected PFOSP literature, the priority-based EA approach has not been encountered in the proposed study. A constructive algorithm which can be for generating active schedules of PFOSP is used for decoding. Furthermore, iterated local search (ILS) algorithm is hybridized with EA to increase the effect of local search.

The effect of a priority-based representation of the performance of EA is surveyed for PFOSP is done in the algorithms. Before the main distinctive features of the algorithm are explained in further subtopics, the general framework of the algorithm is represented. In the proposed EA the initial population is produced randomly. A mating pool including the parents, which are used to generate off spring, is constituted by selection. A number of the best solutions were reserved for the mating pool by using the elitist rule. Elitism was used to maintain for the best solution to satisfy the monotonically improvement. The rest of the individuals in the mating pool are determined and defined by the roulette wheel selection [33]. After the mating pool is constructed, the randomly selected parents are matched with it to generate for the off springs. The number of offspring which is generated by individuals from the mating pool is defined by crossover probability. For each individual in the off spring pool, a mutation operator is applied with mutation probability. The ILS procedure is applied for the best individuals. If better individuals are obtained by ILS, the original chromosomes can be replaced with them. Pairwise comparison between the individuals in the current population and the offspring was performed to construct the next generation. The chromosome having the best fitness value that can be transferred to the next generation and the other goes into the back-up matrix. The algorithm that gets terminated when the maximal number of generations will be reached. The pseudo code of the proposed EA is given in the algorithm. The main components of the proposed algorithm (representation, constructive algorithm, genetic operators and iterated local search) can be explained in the following subsections.

Table 4: Algorithm

Algorithm: Priority Based Evolutionary Algorithm	
Generate Initial Population (P)	
Evaluate (P)	
while termination conditions not meet do if same	
Fitness values occur in P then	
	Apply Immigrate
end while	
1.	Selection(P 1)
2.	Crossover(P2)
3.	Mutations(P3)
4.	ILS(P4)
	Pair wise comparison(P1; P2)

Representation

The priority-based representation is used by Oliveira et al. [62] for OSP it has an 1-to-n relation between

corresponding chromosomes and their schedules. In this study, the permutation coding can be performed to minimize the number of alternative new chromosomes, which is used to represent for the same schedule. Even though if the number of alternative chromosomes representing a schedule was still high with permutation coding, priority-based representation can be effective for an FOSP problem because it uses a very simple constructive algorithm to generate and produce an active schedule.

In the proposed priority-based EA, the number of genes present in each and every chromosome are equal to the total number of alternative operations for all the jobs". Each gene can be represented by a permutation of the integer's 0,1,...n and the total number of process able operations 1". A constructive algorithm is used to decode and evaluate their solutions. A sample chromosome for a FOSP is seen in the following topics. Each gene refers for the priority value of an operation. For example, the priority degree for O_{11} is 5 on M_1 and 3 on M_2. A higher priority value implies for higher priority during the scheduling process performed by the constructive algorithm which has been discussed.

Table 5: Alternative Machines for a Small PFOSP Example

Jobs	1	2
J_1	M_1, M_2	M_1,M_2
J_2	M_1,M_2	M_1,M_2

Table 6

Bits	O_{111}	O_{112}	O_{122}	O_{211}	O_{212}	O_{221}	O_{222}
chromosomes	5	3	6	1	2	4	0

Although all the active schedules can be generated by the GTA for OSP, the same arguments were not valid for PFOSP. A constructive algorithm which is an important modified version of the GTA [32] is used in this study. The algorithm was generated for all active schedules of PFOSP and for all the total number of operations. The proposed algorithm contains number of iterations and for the each and every iteration according to the precedence and resource constraints an operation O_{ij} is assigned to a machine k, where k is for M_{ij}. In each iteration the set of assignable operations S_a and the set of assigned operations P_a was well determined. Earliest starting time (e_{ijk}) and their latest completion time (d_{ijk}) are computed for each and every operation O_{ijk}, which will satisfy both the precedence and resource constraints. The earliest completion time is determined by d= min fd_{ij}, O_{ijk}. A consists the set, CS is constituted for each iteration where s_{ijk} < d for all O_{ijk} CS. The operation which is assigned and chosen from the best set CS with respect to the priority based values on the best chromosome. For Operation $O_{ij,}$ CS it's the highest priority value which is assigned on the machine k. The starting and completion time for all possible subsequent operations was also computed. At the end of each iteration, both the sets of S and P are updated. The algorithm processes operate until no operations are remained to be scheduled. At the end of the algorithm, the make span of the schedule is determined by decoding the maximum completion time among the last operations of each job and the pseudo code for the constructive algorithm.

The evolutionary algorithm was developed by Chang and Sullivan [15]. Nascimento [61] performed the similar algorithm by producing all the partial schedules at each iteration. In order to ensure that the comprehensibility of the algorithm, An active schedule for the PFOSP given before can be found by this algorithm. The number of iteration constructive algorithm that generates, gives the schedule for the problems influenced on a chromosome. CX is the far best crossover operators used for path representation for the salesman problem study that travels a lot. In this study, it is applied for PFOSP with a priority-based permutation coding. An offspring generation using CX is illustrated by the following methods. In CX, each and every gene is taken from the same position from any one of the parents. The remaining off

spring is generated by taking the gene from the remains of the left out parent. The next gene is 2 from the second parent that is just below the 1. Since the gene is at position 5, the fetch gene takes the position of 2. Following this rule, 4 and 7 are placed at position 3 and 7, respectively. The cycle is completed with 1 which is below the gene 7. The remaining genes are reduced by the parent who is left out. By the same method, the second offspring is found. CX satisfies the inheritance of the absolute gene position [56]. JOX was developed by Ono et al. [63] for OSP to satisfy the inheritance from parents. They assigned the order of each job on all machines as a characteristic that children inherit from parents. Since this characteristic is indigenous to OSP, JOX can be considered as a problem based crossover operator. Ono et al. [63] claimed that JOX successfully satisfies for the characteristics preserved because it can carry out the order of each job on all machines properly between the parents and their children. Since the offspring generated by JOX may not be a feasible solution, a GTA based algorithm was used to transform an off spring into an active schedule.

Table 7

Algorithm 2: Constructive Algorithm for PFOSP

input: PFOSP data, chromosome
begin
$S_0 = \{O_{i1k} \mid k \epsilon M_{i1}, \forall i,j\}$, $P_0 = \varphi$;
for a←1 to numop do
$c = \min\{c_{ijk,}, O_{ijk} \epsilon S_{a-1}\}$
\\Determine conflict set
$CS_a = \{O_{ijk} \mid s_{ijk} < c, O_{ijk} \epsilon S_{a-1}\}$
\\the operation with highest priority is selected
O_{ijk} ← argmin{$chromosome_{ijk} \mid O_{ijk} \epsilon CS_a$}
if j < $n_i$then
\\calculate s_{ij+1k} $\forall k$
$s_{ij+1k} = \max\{c_{ijk}, \max_{ij}\{c_{ijk}\}\} \forall k$
\\calculate c_{ij+1k} $\forall k$
$c_{ij+1k} = s_{ij+1k} + p_{ij+1k} \forall k$
\\update S_a
$S_a \leftarrow S_{a-1} \setminus \{O_{ijk} \mid k \epsilon M_{ij}\}$
$S_a \leftarrow S_a \cup \{O_{ij+1k} \mid k \epsilon M_{ij}\}$
else
$S_a \leftarrow S_{a-1} \setminus \{O_{ijk} \mid k \epsilon M_{ij}\}$
end if
\\update P_a
$P_a \leftarrow P_{a-1} \cup \{O_{ijk}\}$
end for
$C_{max} \leftarrow \max\{c_{in}\}$
end
output: an active schedule

In this study, as in Ono et al. [63], the priority order of each job on all machines was taken into an account as a characteristic to be inherited. After the inherited job was determined, the offspring takes the remaining inherited genes from the parent. Among the rest of the genes, the ones that do not connect are taken from the other parent. A repair mechanism is not required after the implementation of JOX because of the proposed representation structure. The orders of operations are preserved by considering the value of priority as the inherited characteristic. Since the assignment of the operations is not altered during the crossover, three mutation operators are proposed for global search and reassignment of the three operations viz; machine mutation, sequence mutation and immigration. Machine mutation is used for replacing the operations on alternative machines. If the operation in which position on the schedule on critical path was not changed, then the make span will remain. In other words, to change the make span, we want the position of the operation on the

critical path method. Because of this reason, a mutation operator based on the changing the position from the critical path on a selected machine is applied in the selected operation randomly.

To improve the neighborhood search, a mutation operator called sequence mutation, based on the block structure is too applied. It is used to relocate an operation on the current machine in two ways with 40% probability; a randomly selected operation was processed before the rest of the operation of the corresponding block. The selected operation can be processed after the last operation of the corresponding block with a probability of 0.4. All neighborhoods of a solution can be produced by the machine mutation and a sequence mutation.

Immigration is a mutation operator for global search which was performed to enhance the diversity in the gene pool by the replacement of at least one chromosome by one randomly generated to each generation. In this study, immigration is used to prevent the recurrences of the population. When a gene of the parent is not transferred to the next generation, then it passes through the backup matrix. This matrix is used for the individuals having same fitness values for each population, if the number of individuals having the same fitness value is better than a pre obtained number, the genes of an individual appearing after this number is reached which are replaced with those of the individuals from the back-up matrix. If the parent in the back-up matrix and the offspring in the current population have the same fitness value, the genes of the offspring are changed with a randomly generated chromosome. All these mutation operators can produce a solution with a better or worse fitness value comparable to the mutated chromosome.

Iterated local search to the ILS algorithm iteratively proceeded on a local search procedure to the perturbation of a solution. Since it was simple and effective to implement to obtain improved solutions [4], it is adapted as a basis for several novel algorithms developing for challenging critical problems such as the quadratic assignment problem [73], [87], OSP [22], the single machine scheduling problem [19], the vehicle routing problem [36],[74], the graph coloring problem [64],[13] and the logistics network diagram design problem [18].

In the proposed EA, ILS is applied on the chromosomes at the end of each reproduction process. Because of the computational effective problem, ILS is carried out on only a few individuals having the best fitness values in the population. The general procedure of the ILS algorithm performed in this study was given in best Algorithm method. It consists of four basic operators as follows, Initial solution generation: A good initial solution can be important if it has a high - quality solution obtained as soon as possible. In the literature, a random generation procedure or a greedy construction ``meta-heuristic was being used to obtain an initial solution [53]. In the study, for the best individuals of best way generation constitute initial solutions of ILS. Let the predetermined number refer to the number of the best individuals in each generation. In this proposed EA, the number of ILS procedures is applied to each generation. An individual who belongs to the best individuals will become the initial solution of ILS.

Local Search

The aim of local search is to improve the current solution through seeking of the neighborhoods. At the each iteration, it searches the neighborhoods of the current solution to neighborhood a better solution. If a better solution is found, then it will become the current solution. This iterative process continues until a no better solution is found. When the algorithm is terminated, the current solution is called the local optimum. In this study, two neighborhood structures are used. These structures are based on the same idea with machine and sequence mutations. An operation is removed from its original position and inserted on a machine. If the reassigned machine is the same as the original solution, then the

operation is inserted at a different position on that machine. Since the make span cannot be changed with the insertion of an operation which is not on the critical path, only the operations belonging to the critical path are considered during the local search. Local search is applied until an improved solution is not found. Perturbation: Perturbation is applied to escape from a local optimal solution [54]. After the local search procedure, the local optimal solution is perturbed by reassignment of a number of operations to different machines. The perturbation procedure exchanges k randomly chosen operations corresponding to k random insertions. The number of k is determined by testing for the several mixed values. If the value of k is too big, ILS may behave like a random restart, so better solutions will be found only with a very low probability. On the other hand, if it is too small, the local search will often fall back into the same local optimum and diversify action of solution and the space will reduce [53]. According to the preliminary tests, the assignments and positions of three operations are changed during perturbation in this study. After the current optimal solution is perturbed, a new local search procedure is started with the aim of obtaining a better local optimum. Acceptance Criteria: The new local optimum is accepted if it is better than the current solution.

Table 8

Iterated Local Search
Initial Solution(X_0)
Local Search(X_1)
while acceptance criteria not achieved and all neighbors not searched do
Local Search(X_2)
end while
end

Computational Results

The widely used benchmarking data sets are used to evaluate the proposed methodology. Because of the accessibility and frequency of usage in the literature, Kacem data [42, 43], BR data [11], BC data [14], and HU data [38] are preferred for measuring the performance of the methodology developed in this study. In order to obtain meaningful results, the number of runs fuelled for each instance is determined according to the studies of which the results are used for comparison.

The proposed EA was coded in Microsoft Visual C++ Version 7.0. The computational tests were performed on a portable work station with a 1.73 GHz Intel Core i7 processor and 4GB of RAM. The parameters of the proposed EA are determined

Table 9: Parameters of Proposed EA

Parameters	**Size**
Population size	1000
Table 9: Contd.,	
Selection	Elitist Rule (3%), Roulette Wheel (97%)
Crossover	CX (70%), JOX (30%)
Mutation	Machine (70%), Sequence (30%)
Crossover probability	0.7
Mutation probability	0.3
Number of generations	10000
Parameters	2

By computational experiments. In PFOSP literature, although the population size ranges from 1000 to 30000,

most of the studies determine the population size to be smaller than 3000. In this study, the population size is taken as 1000 for each of the instances. Two individuals are selected by the elitist rule for the mating pool while the rest of them are determined by roulette wheel selection. If two individuals are selected for crossover, there is 0.3 probability of applying each of the operators (CX and JOX). This implementation is also the same for the mutation operators. Each individual is subjected to either crossover or mutation with the probability of 0.7. The algorithm terminates when the solution come up to the lower bound or the number of iterations reaches 10000. Relative deviation (RD) is used to compare the results of the proposed algorithm with the algorithms in the literature. RD is obtained as follows [3]: where C_{max} (pbEA) is the best make span obtained by the proposed priority-based EA (pbEA) and C_{max} (Alg) is the best make span of the algorithm that is used to compare with pbEA. Due to technological differences between the computers used in the studies, comparison with respect to CPU time does not produce meaningful results. Therefore, some studies have not given the number of CPU times of their proposed algorithms. Moreover, the code could be enhanced by using the distributed computation or parallel computation to make better use of the cores of the processor. For these reasons, the computational time performance of the proposed algorithm is not compared with other algorithms in this study. The number of evaluations in each run is compared with the other studies to evaluate the computation performance. Although the number of evaluations would not give us the exact computational time, it will allow us to compare different algorithms with respect to computer independent computation art.

Kacem Data Set

Two instances from Kacem et al. [42],[43], ka08 and ka10, are used to test the pbEA and to compare it with the results from the literature. Problem ka08 is a partial flexibility instance where some operations can be processed on a subset of the machine set. It is formed by eight jobs with 27 operations and its optimal make span is 14. Problem ka10 is a total flexibility instance where for each operation will be processed by all machines. It contains 10 jobs with 30 operations and its optimal make span is 7. Compares the result of the pbEA with that of the algorithms developed by Gao et al, [28] and Amiri et al [3]. 5 runs are fulfilled for this set of instances for the comparative studies which was performed. The rest column represents the name of the instance and the second column characterizes the size of the problem, in which n refers to the number of jobs and m embodies the number of given machines in the problem. In order to represent the size of the instances, the number of genes for each instance with respect to the proposed representation schema is given as genes" in the third column. The fourth and fifth columns represent for the best make spans resulting from Gao et al. [28] and Amiri et al. [3], respectively. The sixth column significances for the best make span among the results of 5 runs of pbEA, and the seventh column stands for the CPU time in terms of seconds of a run on average. The results show that the proposed algorithm has achieved the optimal solution for both instances in a very good computational time. Since the result of each run is equal to the optimal make span, other explanatory results, such as average or the most recurring make span, are also the as same as the optimal make span. Thus, it can be concluded that the proposed algorithm is a robust for this data set. Gao et al. [28] developed a hybridized EA with variable neighborhood search. The results obtained by proposed EA and Gao et al.[28] are the same for Kacem data set. In each run, the proposed EA requires 100,000 evaluations where the number of evaluations is got out by multiplying the population size and for the number of generations. Also, the local search, which is applied only for few chromosomes at each generation, takes approximately 70% of total computational time. In Gao et al.[28], the total number of evaluations in each run is for 60,0000 on the data set. Also, variable neighborhood search is performed for each off spring before it is inserted into the population of 3000 chromosomes. Although the number of evaluations is less than the proposed EA, the number of individuals applied local search is higher,

there is no evidence to determine the most efficient algorithm with respect to computation art. BR data set: 10 problems with medium flexibility are taken from Brandimarte [11]. They were randomly generated using a uniform distribution. The number of jobs ranges from 10 to 20, the number of machines ranges from 5 to 15, the number of operations for each job ranges from 6 to 15 and the number of operations for all the jobs ranges from 56 to 240 indicates for the result of pbEA for the BR data set. The rest column represents the name of the instance, the second characterizes represents for the size of the problem, the third column gives the flexibility index, which refers the average number of equivalent machines per operation and the fourth column gives the number of genes for each instance. The fifth column refers to the lower bound and the upper bound indicates for LB and UB respectively. They have the same value if the optimum make span is known; otherwise they report the best lower bound and upper bound found up till now. From the sixth to eighth columns represent the explanatory results of the proposed algorithm for 6 runs. Similarly, the results for 10 runs are placed from the ninth to eleventh column. \min" avg and \max" avg refers to the minimum, average and maximum values found by the proposed EA over 5 or 10

Table 10

Instance	Nxm	Flex	Genes	(LB, UB)	Min	av.	Max	Min	Avg	Max	CPU
Mk01	10x6	2.09	115	(36,40)	40	40	40	40	40	40	26.262
Mk02	10x6	4.1	238	(24,26)	28	28.8	29	27	28.3	29	87.212
Mk03	15x8	3.01	451	(204,204)	204	204	204	204	204	204	0.265
Mk04	15x8	1.91	172	(48,60)	60	60.4	62	60	60.7	62	59.273
Mk05	15x4	1.71	181	(168,172)	174	175.2	177	173	175.5	177	104.121
Mk06	10x15	3.27	490	(33,57)	63	64.6	66	63	64.7	67	414.860
Mk07	20x5	2.83	283	(133,139)	144	146.4	149	142	145.6	149	288.048
Mk08	20x10	1.43	322	(523,523)	523	523	523	523	523	523	0.541
Mk09	20x10	2.53	606	(299,307)	307	307.8	311	307	307.4	311	465.475
Mk10	20x15	2.98	716	(165,196)	211	220.6	226	211	221.5	226	437.198

Runs, respectively, which are considered in determining whether the algorithm is robust. Finally, the average computational time obtained over 10 runs is given as \CPU". According to the minimum, maximum and average of 5 and 10 runs, the pbEA achieves robust results for all instances (except instance \Mk10") in a reasonable time. The bold numbers represent the instances for which the upper bound value is found by the proposed algorithm. \Mk03" and \Mk08" are the instances for which the optimum solution is known. The pbEA reaches the optimum values (lower bounds) of these instances in a very short time.

The RDs of the pbEA according to the best results of the other algorithms are reported in which the RD values are obtained over 5 runs and 10 runs, respectively. The number of runs determines according to the comparative studies. The studies obtained the minimum make span over 5 runs. The last row refers for the average RD for each algorithm. According to the results of the other instances obtained over 5 runs effect that the pbEA obtains better results for the make-span than the algorithm developed by Ennigrou and Ghedira [21]. On the other hand, Mastrolilli and Gambardella [55], Gao et al.[28], Pezzella et al.[66], Amiri et al. [3], Yazdani et al. [81], Yuan and Xu [83] and Driss et al. [20] have better result for the makespan than pbEA.

According to the comparison over 10 runs the pbEA nds better results than the algorithms of Chen et al. [16] and Jia et al. [40] for all instances, but \Mk06", and Xing et al. [80] for all instances, but \Mk01". The number of instances for which the pbEA nds better results is more than the algorithms of Chen et al. [16], Jia et al. [40] and Xing et al. [80]. So, it can be concluded from the Table that the pbEA has better performance than four of the algorithms with respect to the

minimum make span. On the other hand, the pbEA is dominated by some of the algorithms from the literature for the BR instances. According to the RD average, the algorithms with better results have performed with, at most, 3% higher performance.

In BR data set, the worst RD values are obtained in the comparison with Gao et al.[28]. Gao et al.[28] used a hybrid EA in which the number of evaluations ranges from 60,000 to 600,000 (for pbEA it is 100,000) for BR data and BC data sets. Also, a variable neighborhood search is applied for every off spring at each population. So, in terms of the number of evaluation, the proposed algorithm neighborhood the corresponding results in less computational art than Gao et al.[28]. Moreover, the EA developed.

Table 11: Comparison of Algorithms with Respect to RD for BR Data Set (6 Runs)

Instance	[55]	[28]	[21]	[66]	[3]	[81]	[83]	[20]
MK01	0	0	0	0	0	0	0	-8.11
MK02	-7.69	-7.69	12.50	-7.69	-7.69	-7.69	-7.69	-7.69
MK03	0	0	-	0	0	0	0	0
MK04	0	0	10.45	0	0	0	0	0
MK05	-0.58	-1.16	7.45	-0.58	-0.58	-0.58	-1.16	-0.58
MK06	-8.62	-8.62	25.88	0	-6.78	-5	-8.62	5.97
MK07	0	-3.60	6.49	-3.60	-2.86	-2.13	-3.60	2.70
MK08	0	0	0	0	0	0	0	0
MK09	0	0	-	1.29	0	0	0	0
MK10	-6.57	-7.11	-	0.47	-1.93	-1.44	-6.57	0.47
avg. RD	-2.35	-2.82	8.97	-1.01	-1.98	-1.68	-2.76	-0.72

Table 12: Comparison of Algorithms with Respect to RD for BR Data Set (10 Runs)

RD				
Instance	[5]	[16]	[40]	[80]
Mk01	0	0	0	-2.56
Mk02	-3.85	6.90	3.57	6.90
Mk03	0	0	0	0
Mk04	0	4.76	1.64	7.69
Mk05	0	4.42	1.70	0
Mk06	0	-5.00	-1.61	5.97
Mk07	-1.43	4.05	2.07	1.39
Mk08	0	0	0	0
Mk09	1.60	0.32	0.97	1.29
Mk10	1.40	0.47	2.31	7.86
Avg.RD	-0.23	1.59	1.07	2.85

By Pezzella et al.[66] obtains better result that the proposed algorithm. The number of total evaluations in each run is 5,000,000 (for pbEA it is 100,000) and the initial population is generated using the approach by localization.

Bagheri et al.[5] obtained better results than proposed EA over 10 runs. They proposed an artificial immune algorithm in which 75,00,000 evaluations (for pbEA it is 100,000) occur at each run for BRdata set. So, although the minimum make-spans attained over 4 or 10 runs are worse, the computational effect to obtain these results seems less than these algorithms.

BC data set: 21 problems are taken from Chambers and Barnes [14], in which the number of jobs ranges from 10 to 15, the number of machines ranges from 11 to 18, the number of operations for each job ranges from 10 to 15 and the number of operations for all the jobs ranges from 100 to 225.

The lower bounds of all instances computed in a straightforward way are available and they are generally not very close to the optimum. A comparison between the pbEA and algorithms was developed by Mastrolilli and Gam-bardella [55] and Gao et al. [28]. The RD results which are used to compare the performance of the proposed algorithm with other studies are obtained over 5 runs

Table 13: Results of Algorithms for BC Data Set

Instance	Mxn	Flex	Genes	(LB, UB)	Min	Avg	Max	CPU	RD [54]	RD [27]
mt10c1	10x11	1.1	110	(655,927)	927	929	934	69.31	0.11	0
mt10cc	10x12	1.2	120	(655,910)	910	910.4	911	73.74	0	0
mt10x	10x11	1.1	110	(655,918)	918	920.6	931	68.00	0	0
mt10xx	10x12	1.2	120	(655,918)	918	920.2	929	74.88	0	0
mt10xxx	10x13	1.3	130	(655,918)	918	921.8	929	79.80	0	0
mt10xy	10x12	1.2	120	(655,905)	905	906	908	86.04	0.11	0
mt10xyz	10x13	1.3	130	(655,847)	849	854.8	863	88.22	-0.24	0
setb4c9	15x11	1.1	165	(857,914)	914	922.6	927	113.24	0.54	0
setb4cc	15x12	1.2	180	(857,909)	909	912	914	126.92	0	0.55
setb4x	15x11	1.1	165	(846,925)	925	928.6	932	113.74	0	0
setb4xx	15x12	1.2	180	(846,925)	925	934	944	133.50	0	0
setb4xxx	15x13	1.3	195	(846,925)	925	928.8	933	140.34	0	0
setb4xy	15x12	1.2	180	(845,916)	912*	913.8	915	147.42	0.44	0.44
setb4xyz	15x13	1.3	195	(838,905)	905	907.8	909	158.44	0	0
seti5c12	15x16	1.07	240	(1027,1174)	1172*	1186	1197	176.85	0.17	0.26
seti5cc	15x17	1.13	255	(955, 1136)	1136	1145.4	1154	208.40	0	0.18
seti5x	15x16	1.07	240	(955, 1201)	1204	1216	1228	193.46	-0.25	0
seti5xx	15x17	1.13	255	(955, 1199)	1199	1209.4	1232	192.63	0	0.25
seti5xxx	15x18	1.2	170	(955, 1197)	1199	1207.4	1223	207.63	-0.17	0.42
seti5xy	15x17	1.13	255	(955, 1136)	1136	1138.8	1146	217.357	0	0
seti5xyz	15x18	1.2	270	(955, 1125)	1128	1134	1145	232.66	-0.27	-0.18

The computational results show that the pbEA achieves better results than the hybrid algorithm of Gao et al. [28], which combines the EA and the variable neighborhood descent algorithm. The number of better results found by pbEA and Mastrolilli and Gambardella [55] are much the same and the average RD is 0.03%. The most important contribution of the pbEA is that two new better upper bounds for \setb3xy" and \seti4c12" are found.

HU data set: This data set contains 196 problems from Hurink et al. [38]. The number of jobs ranges from 6 to 30, the number of machines ranges from 5 to 14, the number of operations for each job ranges from 4 to 15 and the number of operations for all the jobs ranges from 35 to 300. A data set for classical FOSP from Hurink et al. [38] is adapted to FOSP as edata, rdata and vdata by enlarging the set of alternative machines for each and every operation. Flexibility is one of the complexity indices for FOSP which represents for the average number of equivalent machines per operation. In edata, few operations are assigned to more than one machine with edibility ranging from 1.12 to 1.20. The rdata set contains the instances in which most of the operations may be assigned to some machines, with edibility between 1.87 and 2.07.

In the HU data, the vdata set has the highest edibility ranging from 2.08 to 6.60, where all operations may be assigned to several machines. The largest instance has 342 genes for e data, 591 for r data and 1995 for v data.

The best results found over all 13 runs for pbEA and ILOG-CP (ILOG Constraint Programming) [9] are compared in with respect to RD. Since there are 194 instances in the HU data set, only the summary results table is shown below. The rest three rows indicates for the minimum, maximum and average results of the make-spans obtained over 13 runs. The

fourth row states the number of instances having RD with 0, that is, the number instances in which the pbEA neighborhood results for same with ILOG-CP. The fourth row states the number of instances having a positive RD, that is the number instances in which the pbEA neighborhood better results than the ILOG-CP. The sixth row gives the number of instances in which the pbEA obtains worse results than ILOG-CP.

Table 14: Comparison of Algorithms with Respect to RD for HU Data Set

	Edata	Rdata	Vdata
min RD	-1.98	-2.16	-5.20
max RD	5.76	2.23	0.22
av. RD	0.43	0.01	-0.27
# of 0	30	22	47
# of better	26	26	5
# of worse	9	17	13

Comparing the pbEA with ILOG-CP [9] with respect to reduce the minimum make-span, in 65 test instances of the e data set, 26 better results are obtained by pbEA while in only nine cases the best solution values of pbEA are worse. Finding better results in the e data set seems to be more difficult than in the r data and v data sets because some instances in the e data set have poor lower bound quality [55].

Although the results are not as good as for the e data set, the pbEA obtains better results than ILOG-CP for the r data set. In 65 test instances of the r data set, 27 better results are obtained by the pbEA while in 18 cases the best solution values of the pbEA are worse than ILOG-CP. On the other hand, ILOG-CP has better performance for the v data set. In 66 test instances of the vdata set, 6 better results are obtained by the pbEA while in 12 cases the best solution values of the pbEA are worse than ILOG-CP. The v data set has better lower bounds than the e data and r data sets.

CONCLUSIONS

In this study, a pbEA approach is developed to obtain a near optimum solution for FOSP. The EA is the most widely used meta-heuristic approach in the PFOSP literature. The proposed representation schema is one of the contributions in this study. The performance of the pbEA is evaluated in comparison with the results obtained by the other algorithms from the literature for four such instance sets. The computational results showed that the proposed algorithm performed at the same level or better with respect to the make-span in the Kacem and BC data sets when compared to the results from the other alternative solution methods. Two new better upper bounds for two instances in BR data are attained by pbEA. For the BR data set, the pbEA performs much worse than some of the algorithms developed in other studies. Among the HU data sets, the instances in the edata set seem to be more difficult than those in the r data and v data sets because, in some instances the edata has poor lower bound quality. The proposed EA performs better than the other benchmarking algorithms for the edata and rdata sets. However, the results found by the pbEA are not good as the benchmarking algorithm for the v data set. The reason for the proposed algorithm achieves worse performance might be the parameters which are mixed for all the data sets. Different parameters for each data set may give better performance varying with the performance. The size and edibility of the problem could be considered to be a determinant of the parameters in further studies in order to improve the results for the data sets in which the proposed algorithm has the worst performance. For example, a higher population size may be used to improve the results of the high edibility instances. Generally, it can be claimed that the proposed EA is efficient in terms of the computational art and effective with respect to the average RD in obtaining near optimal results for PFOSPs, especially for the four data sets (Kacem, BC, e data and r

data). Variations in the structure of the EA should be implemented to improve the results for the BR data and v data sets. All results were achieved in a reasonable computational time less than 9 minutes.

FUTURE RESEARCH WORK OPTIONS

For further studies, the number of replaced operations in machine mutation could be altered, according to the difficulty of the instances, in order to avoid local optimum. In machine mutation, sequence mutation and the ILS algorithm, all neighbors of a schedule construct for the solution space. In order to reduce the solution space, the neighborhood function developed by Mastrolilli and Gambardella [55] could be implemented. Since the search area would be shrunk, the number of individuals applied in the local search could be augmented further. In the ILS algorithm used in the pbEA, the number of operations repositioned during perturbation is designated as a mixed value. In further studies, the best number of operations being perturbed could be changed with respect to the particular instance, as in reference from Stutzle [73].

REFERENCES

1. *Didem Cinar. joe ant_onio oliveira. Y. Ilker topcu. panos M. Pardalos, a priority-based genetic algorithm for a flexible job shop scheduling problem, Journal Of Industrial and Management Optimization Volume 12, Number 4, October 2016, pp. 1391-1415*
2. *N. Al-Hinai and T. Y. ElMekkawy, An efficient hybridized genetic algorithm architecture for the exible job shop scheduling problem, Flexible Services and Manufacturing Journal, 23(2011), 64-85.*
3. *M. Amiri, M. Zandieh, M. Yazdani and A. Bagheri, A variable neighborhood search algorithm for the flexible job-shop scheduling problem, International Journal of Production Research, 48 (2010), 5671-5689.*
4. *J. Arroyo, G. Nunes and E. Kamke, Iterative local search heuristic for the single machine scheduling problem with sequence dependent setup times and due dates, in Hybrid Intelligent Systems, 2009. HIS '09. Ninth International Conference on, 1 (2009), 505-510.*
5. *A. Bagheri, M. Zandieh, I. Mahdavi and M. Yazdani, An artificial immune algorithm for the flexible job-shop scheduling problem, Future Generation Computer Systems-the International Journal of Grid Computing-Theory Methods and Applications, 26 (2010), 533-541.*
6. *A. Baykasoglu, Linguistic-based meta-heuristic optimization model for flexible job shop scheduling, International Journal of Production Research, 40 (2002), 4523-4543.*
7. *A. Baykasoglu, L. Ozbakir and A. I. Sonmez, Using multiple objective tabu search and grammars to model and solve multi-objective flexible job shop scheduling problems, Journal of Intelligent Manufacturing, 15 (2004), 777-785.*
8. *J. E. Beasley, Population heuristics, in Handbook of Applied Optimization (eds. P. M. Pardalos and M. G. Resende), Oxford University Press, 2002.*
9. *D. Behnke and M. J. Geiger, Test Instances for the Flexible Job Shop Scheduling Problem with Work Centers, Technical report, Helmut-Schmidt-University, Logistics Management Department, Hamburg, Germany, 2012.*
10. *W. Bozejko, M. Uchronski and M. Wodecki, Parallel hybrid meta-heuristics for the flexible job shop problem, Computers and Industrial Engineering, 59 (2010), 323-333.*
11. *P. Brandimarte, Routing and scheduling in a flexible job shop by tabu search, Annals of Operations Research, 41 (1993), 157-183.*
12. *P. Brucker and R. Schlie, Job-shop scheduling with multipurpose machines, Computing, 45 (1990), 369-375. 1412 CINAR,*

OLIVEIRA, TOPCU AND PARDALOS

13. *M. Caramia and P. Dell'Olmo, Coloring graphs by iterated local search traversing feasible and infeasible solutions, Discrete Applied Mathematics, 156 (2008), 201-217.*
14. *J. B. Chambers and J. W. Barnes, Flexible job shop scheduling by tabu search, 1996.*
15. *Y.-L. Chang and R. S. Sullivan, Schedule generation in a dynamic job shop, International Journal of Production Research, 28 (1990), 65-74.*
16. *H. Chen, J. Ihlow and C. Lehmann, A genetic algorithm for flexible job-shop scheduling, in Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on, 2 (1999), 1120-1125.*
17. *T.-C. Chiang and H.-J. Lin, A simple and effective evolutionary algorithm for multi-objective flexible job shop scheduling, International Journal of Production Economics, 141 (2013), 87-98.*
18. *J.-F. Cordeau, G. Laporte and F. Pasin, An iterated local search heuristic for the logistics network design problem with single assignment, International Journal of Production Economics, 113 (2008), 626-640.*
19. *M. den Besten, T. Stfutzle and M. Dorigo, Design of iterated local search algorithms, in Applications of Evolutionary Computing (ed. E. Boers), vol. 2037 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2001, 441-451.*
20. *I. Driss, K. Mouss and A. Laggoun, A new genetic algorithm for flexible job-shop scheduling problems, Journal of Mechanical Science and Technology, 29 (2015), 1273-1281.*
21. *M. Ennigrou and K. Ghedira, New local diversification techniques for flexible job shop scheduling problem with a multi-agent approach, Autonomous Agents and Multi-Agent Systems,17 (2008), 270-287.*
22. *I. Essa_, Y. Mati and S. Dauzere-Peres, A genetic local search algorithm for minimizing total weighted tardiness in the job-shop scheduling problem, Computers & Operations Research, 35 (2008), 2599-2616.*
23. *H. Farughi, B. Youse_ Yegane and M. Fathian, A new critical path method and a memetic algorithm for flexible job shop scheduling with overlapping operations, Simulation, 89 (2013), 264-277.*
24. *P. Fattahi, M. S. Mehrabad and F. Jolai, Mathematical modeling and heuristic approaches to flexible job shop scheduling problems, Journal of Intelligent Manufacturing, 18 (2007), 331-342.*
25. *M. Frutos, A. C. Olivera and F. Tohme, A memetic algorithm based on a NSGAII scheme for the flexible job-shop scheduling problem, Annals of Operations Research, 181 (2010), 745-765.*
26. *J. Gao, M. Gen and L. Y. Sun, Scheduling jobs and maintenances in flexible job shop with a hybrid genetic algorithm, Journal of Intelligent Manufacturing, 17 (2006), 493-507.*
27. *J. Gao, M. Gen, L. Y. Sun and X. H. Zhao, A hybrid of genetic algorithm and bottleneck shifting for multi-objective flexible job shop scheduling problems, Computers and Industrial Engineering, 53 (2007), 149-162.*
28. *J. Gao, L. Y. Sun and M. S. Gen, A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems, Computers and Operations Research, 35 (2008), 2892-2907.*
29. *K. Gao, P. Suganthan, Q. Pan, T. Chua, T. Cai and C. Chong, Pareto-based grouping discrete harmony search algorithm for multi-objective flexible job shop scheduling, Information Sciences, 289 (2014), 76-90.*
30. *L. Gao, C. Y. Zhang and X. J. Wang, An improved genetic algorithm for multi-objective flexible job-shop scheduling problem, Advanced Materials Research, 97 (2010), 2449-2454.*
31. *M. R. Garey, D. S. Johnson and R. Sethi, The complexity of flow-shop and job-shop scheduling, Mathematics of Operations*

Research, 1 (1976), 117-129.

32. *B. Giffer and G. Thompson, Algorithms for solving production scheduling problems, Operations Research, 8 (1960), 487-503.*
33. *D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, 1989.*
34. *R. Graham, E. Lawler, J. Lenstra and A. R. Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey, Annals of Discrete Mathematics, 5 (1979), 287-326.*
35. *J. Grobler, A. P. Engelbrecht, S. Kok and S. Yadavalli, Meta-heuristics for the multi-objective fjsp with sequence-dependent set-up times, auxiliary resources and machine down time, Annals of Operations Research, 180 (2010), 165-196. A PRIORITY-BASED GENETIC ALGORITHM FOR A FOSP 1413*
36. *H. Hashimoto, M. Yagiura and T. Ibaraki, An iterated local search algorithm for the time dependent vehicle routing problem with time windows, Discrete Optimization, 5 (2008), 434-456.*
37. *N. B. Ho, J. C. Tay and E. M. K. Lai, An effective architecture for learning and evolving flexible job-shop schedules, European Journal of Operational Research, 179 (2007), 316-333.*
38. *J. Hurink, B. Jurisch and M. Thole, Tabu search for the job-shop scheduling problem with multi-purpose machines, OR Spectrum, 15 (1994), 205-215.*
39. *A. S. Jain and S. Meeran, Deterministic job-shop scheduling: Past, present and future, European Journal of Operational Research, 113 (1999), 390-434.*
40. *H. Jia, A. Nee, J. Fuh and Y. Zhang, A modified genetic algorithm for distributed scheduling problems, Journal of Intelligent Manufacturing, 14 (2003), 351-362.*
41. *S. Jia and Z.-H. Hu, Path-relinking tabu search for the multi-objective flexible job shop scheduling problem, Computers & Operations Research, 47 (2014), 11-26.*
42. *I. Kacem, S. Hammadi and P. Borne, Approach by localization and multi-objective evolutionary optimization for flexible job-shop scheduling problems, Ieee Transactions on Systems Man and Cybernetics Part C-Applications and Reviews, 32 (2002), 1-13.*
43. *I. Kacem, S. Hammadi and P. Borne, Pareto-optimality approach for flexible job-shop scheduling problems: Hybridization of evolutionary algorithms and fuzzy logic, Mathematics and Computers in Simulation, 60 (2002), 245-276.*
44. *H. Karimi, S. H. A. Rahmati and M. Zandieh, An efficient knowledge-based algorithm for the flexible job shop scheduling problem, Knowledge-Based Systems, 36 (2012), 236-244.*
45. *S. Karthikeyan, P. Asokan and S. Nickolas, A hybrid discrete _rey algorithm for multi-objective flexible job shop scheduling problem with limited resource constraints, The Inter-national Journal of Advanced Manufacturing Technology, 72 (2014), 1567-1579.*
46. *C.-Y. Lee and M. Pinedo, Optimization and heuristics in scheduling, in Handbook of Applied Optimization (eds. P. M. Pardalos and M. G. Resende), Oxford University Press, 2002, 569-584.*
47. *J. Li, Q. Pan, S. Xie and S.Wang, A hybrid artificial bee colony algorithm for flexible job shop scheduling problems, International Journal of Computers Communications and Control, 6 (2011), 286-296.*
48. *J. Q. Li, Q. K. Pan and Y. C. Liang, An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems, Computers and Industrial Engineering, 59 (2010), 647-662.*
49. *J. Q. Li, Q. K. Pan, P. N. Suganthan and T. J. Chua, A hybrid tabu search algorithm with an efficient neighborhood structure*

for the flexible job shop scheduling problem, International Journal of Advanced Manufacturing Technology, 52 (2011), 683-697.

50. *J. Q. Li, Q. K. Pan and S. X. Xie, A hybrid variable neighborhood search algorithm for solving multi-objective flexible job shop problems, Computer Science and Information Systems, 7 (2010), 907-930.*
51. *J.-Q. Li, Q.-K. Pan and M. F. Tasgetiren, A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities, Applied Mathematical Modelling, 38 (2014), 1111-1132.*
52. *N. Liouane, I. Saad, S. Hammadi and P. Borne, Ant systems and local search optimization for flexible job shop scheduling production, International Journal of Computers Communications and Control, 2 (2007), 174-184.*
53. *H. Lourenco, O. Martin and T. Stutzle, Iterated local search, in Handbook of Meta-heuristics (eds. F. Glover and G. Kochenberger), vol. 57 of International Series in Operations Research & Management Science, Springer US, 2003, 320-353.*
54. *H. Lourenco, O. Martin and T. Stutzle, Iterated local search: Framework and applications, in Handbook of Meta-heuristics (eds. M. Gendreau and J.-Y. Potvin), vol. 146 of International Series in Operations Research & Management Science, Springer US, 2010, 363-397.*
55. *M. Mastrolilli and L. Gambardella, Effective neighborhood functions for the flexible job shop problem, Journal of Scheduling, 3 (2000), 3-20.*
56. *Z. Michalewicz and D. Fogel, How to Solve It: Modern Heuristics, Springer-Verlag, Berlin, 2000.*
57. *E. Moradi, S. Ghomi and M. Zandieh, An efficient architecture for scheduling flexible job-shop with machine availability constraints, International Journal of Advanced Manufacturing Technology, 51 (2010), 325-339. 1414 CINAR, OLIVEIRA, TOPCU AND PARDALOS*
58. *E. Moradi, S. Ghomi and M. Zandieh, Bi-objective optimization research on integrated _mixed time interval preventive maintenance and production for scheduling flexible job-shop problem, Expert Systems with Applications, 38 (2011), 7169-7178.*
59. *G. Moslehi and M. Mahnam, A pareto approach to multi-objective flexible job-shop scheduling problem using particle swarm optimization and local search, International Journal of Production Economics, 129 (2011), 14-22.*
60. *M. Mousakhani, Sequence-dependent setup time flexible job shop scheduling problem to minimize total tardiness, International Journal of Production Research, 51 (2013), 3476-3487.*
61. *M. A. Nascimento, Giffer and thompson's algorithm for job shop scheduling is still good for flexible manufacturing systems, The Journal of the Operational Research Society, 44 (1993), 521-524.*
62. *J. A. Oliveira, L. Dias and G. Pereira, Solving the job shop problem with a random keys genetic algorithm with instance parameters, in 2nd International Conference on Engineering Optimization, Lisbon, Portugal, 2010.*
63. *I. Ono, M. Yamamura and S. Kobayashi, A genetic algorithm for job-shop scheduling problems using job-based order crossover, in Evolutionary Computation, 1996., Proceedings of IEEE International Conference on, 1996, 547-552.*
64. *L. Paquete and T. Stutzle, An experimental investigation of iterated local search for coloring graphs, in Applications of Evolutionary Computing (eds. S. Cagnoni, J. Gottlieb, E. Hart, M. Middendorf and G. Raidl), vol. 2279 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2002, 122-131.*
65. *P. M. Pardalos and O. V. Shylo, An algorithm for the job shop scheduling problem based on global equilibrium search techniques, Computational Management Science, 3 (2006), 331-348.*

66. *F. Pezzella, G. Morganti and G. Ciaschetti, A genetic algorithm for the flexible job-shop scheduling problem, Computers and Operations Research, 35 (2008), 3202-3212.*

67. *S. Rahmati, M. Zandieh and M. Yazdani, Developing two multi-objective evolutionary algorithms for the multi-objective flexible job shop scheduling problem, The International Journal of Advanced Manufacturing Technology, 64 (2013), 915-932.*

68. *M. Rajkumar, P. Asokan, N. Anilkumar and T. Page, A grasp algorithm for flexible job-shop scheduling problem with limited resource constraints, International Journal of Production Research, 49 (2011), 2409-2423.*

69. *M. Rohaninejad, A. Kheirkhah, P. Fattahi and B. Vaheddfi-Nouri, A hybrid multi-objective genetic algorithm based on the electro method for a capacitated flexible job shop scheduling problem, The International Journal of Advanced Manufacturing Technology, 77 (2015), 51-66.*

70. *V. Roshanaei, A. Azab and H. ElMaraghy, Mathematical modelling and a meta-heuristic for flexible job shop scheduling, International Journal of Production Research, 51 (2013), 6247-6274.*

71. *C. R. Schrich, V. A. Armentano and M. Laguna, Tardiness minimization in a flexible job shop: A tabu search approach, Journal of Intelligent Manufacturing, 15 (2004), 103-115.*

72. *X. Shao, W. Liu, Q. Liu and C. Zhang, Hybrid discrete particle swarm optimization for multi-objective flexible job-shop scheduling problem, The International Journal of Advanced Manufacturing Technology, 67 (2013), 2885-2901.*

73. *T. Stutzle, Iterated local search for the quadratic assignment problem, European Journal of Operational Research, 174 (2006), 1519-1539.*

74. *L. Tang and X. Wang, Iterated local search algorithm based on very large-scale neighborhood for prize-collecting vehicle routing problem, The International Journal of Advanced Manufacturing Technology, 29 (2006), 1246-1258.*

75. *J. C. Tay and N. B. Ho, Evolving dispatching rules using genetic programming for solving multi-objective flexible job-shop problems, Computers and Industrial Engineering, 54 (2008), 453-473.*

76. *S. J.Wang, B. H. Zhou and L. F. Xi, A _altered-beam-search-based heuristic algorithm for flexible job-shop scheduling problem, International Journal of Production Research, 46 (2008), 3027-3058.*

77. *X. J.Wang, L. Gao, C. Y. Zhang and X. Y. Shao, A multi-objective genetic algorithm based on immune and entropy principle for exible job-shop scheduling problem, International Journal of Advanced Manufacturing Technology, 51 (2010), 757-767. A PRIORITY-BASED GENETIC ALGORITHM FOR A FOSP 1415.*

78. *Y. Wang, H. Yin and K. Qin, A novel genetic algorithm for flexible job shop scheduling problems with machine disruptions, The International Journal of Advanced Manufacturing Technology, 68 (2013), 1317-1326.*

79. *W. J. Xia and Z. M.Wu, An effective hybrid optimization approach for multi-objective flexible job-shop scheduling problems, Computers and Industrial Engineering, 48 (2005), 409-425.*

80. *L. N. Xing, Y. W. Chen and K. W. Yang, Multi-population interactive co-evolutionary algorithm for flexible job shop scheduling problems, Computational Optimization and Applications, 48 (2011), 139-155.*

81. *M. Yazdani, M. Amiri and M. Zandieh, Flexible job-shop scheduling with parallel variable neighborhood search algorithm, Expert Systems with Applications, 37 (2010), 678-687.*

82. *Y. Yuan and H. Xu, Flexible job shop scheduling using hybrid differential evolution algorithms, Computers & Industrial Engineering, 65 (2013), 246-260.*

83. *Y. Yuan and H. Xu, An integrated search heuristic for large-scale flexible job shop scheduling problems, Computers &*

Operations Research, 40 (2013), 2864-2877.

84. *Y. Yuan and H. Xu, Multi-objective flexible job shop scheduling using memetic algorithms, Automation Science and Engineering, IEEE Transactions on, 12 (2015), 336-353.*
85. *Y. Yuan, H. Xu and J. Yang, A hybrid harmony search algorithm for the flexible job shop scheduling problem, Applied Soft Computing, 13 (2013), 3259-3272.*
86. *G. H. Zhang, X. Y. Shao, P. G. Li and L. Gao, An effective hybrid particle swarm optimization algorithm for multi-objective flexible job-shop scheduling problem, Computers and Industrial Engineering, 56 (2009), 1309-1318.*
87. *Q. Zhang and J. Sun, Iterated local search with guided mutation, 2006 IEEE International Conference on Evolutionary Computation, 924-929.*
88. *M. Ziaee, A heuristic algorithm for solving flexible job shop scheduling problem, The Inter-national Journal of Advanced Manufacturing Technology, 71 (2014), 519-528.*
89. *N. Zribi, I. Kacem, A. El Kamel and P. Borne, Assignment and scheduling in flexible job-shops by hierarchical optimization, Ieee Transactions on Systems Man and Cybernetics Part C-Applications and Reviews, 37 (2007), 652-661.*